TRACKING-LEARNING-DETECTION

V. H. AYMA

Laboratório de Visão Computacional
Departamento de Engenharia Elétrica
PUC – Rio

CONTENT

1. INTRODUCTION
2. TLD FRAMEWORK
    - TRACKING
    - DETECTION
    - LEARNING

CONTENT

1. INTRODUCTION
2. TLD FRAMEWORK
3. TRACKING
4. DETECTION
5. LEARNING

INTRODUCTION

- TLD Framework[1] was proposed by Zdenek Kalal, Krystian Mikolajczyk and Jiri Matas.
  - P-N Learning: Bootstrapping Binary Classifiers by Structural Constrains[2].
  - Forward-Backward Error: Automatic Detection of Tracking Failures[3].

1) Z. Kalal, K. Mikolajczyk and J. Matas, "Tracking-Learning-Detection", Pattern Analysis and Machine Intelligence, 2011.
2) Z. Kalal, K. Mikolajczyk and J. Matas, "Forward-Backward Error: Automatic Detection of Tracking Failures", International Conference on Pattern Recognition, 2010, pp. 23-26.
3) Z. Kalal, J. Matas and K. Mikolajczyk, "P-N Learning: Bootstrapping Binary Classifiers by Structural Constrains", Conference on Computer Vision and Pattern Recognition, 2010.

INTRODUCTION
LONG TERM TRACKING

The goal is to determine the object's bounding box or indicate the object is not visible in the frames that fallows.

INTRODUCTION
LONG TERM TRACKING

Long-term tracking algorithms must:

- Handle:
  - Scale variations.
  - Illumination variations.
  - Occlusions.
  - Background clutter.

- Operate at frame rate (real time).

INTRODUCTION
LONG TERM TRACKING

Long-term tracking algorithms must:

- Detect the object when it reappears in the camera's field of view.

- Handle:
  - Scale variations.
  - Illumination variations.
  - Occlusions.
  - Background clutter.

- Operate at frame rate (real time).

INTRODUCTION
LONG TERM TRACKING

Long-term tracking algorithms must:

- Detect the object when it reappears in the camera's field of view.
  - Object might change its appearance during its absence, thus initial appearance becomes irrelevant.

- Handle:
  - Scale variations.
  - Illumination variations.
  - Occlusions.
  - Background clutter.

- Operate at frame rate (real time).

INTRODUCTION
LONG TERM TRACKING

LONG-TERM TRACKING

TRACKING

(Estimates location)

DETECTION

(Finds the best match)

LONG-TERM TRACKING

TRACKING

(Estimates location)

DETECTION

(Finds the best match)

+ Requires initialization
+ Produces smooth trajectories
+ Reasonably fast

– Accumulate errors during track (drifts)
– Fails when the object disappears
– Doesn't have a post failure recovery mechanism

LONG-TERM TRACKING

TRACKING

(Estimates location)

+ Requires initialization
+ Produces smooth trajectories
+ Reasonably fast

– Accumulate errors during track (drifts)
– Fails when the object disappears
– Doesn't have a post failure recovery mechanism

DETECTION

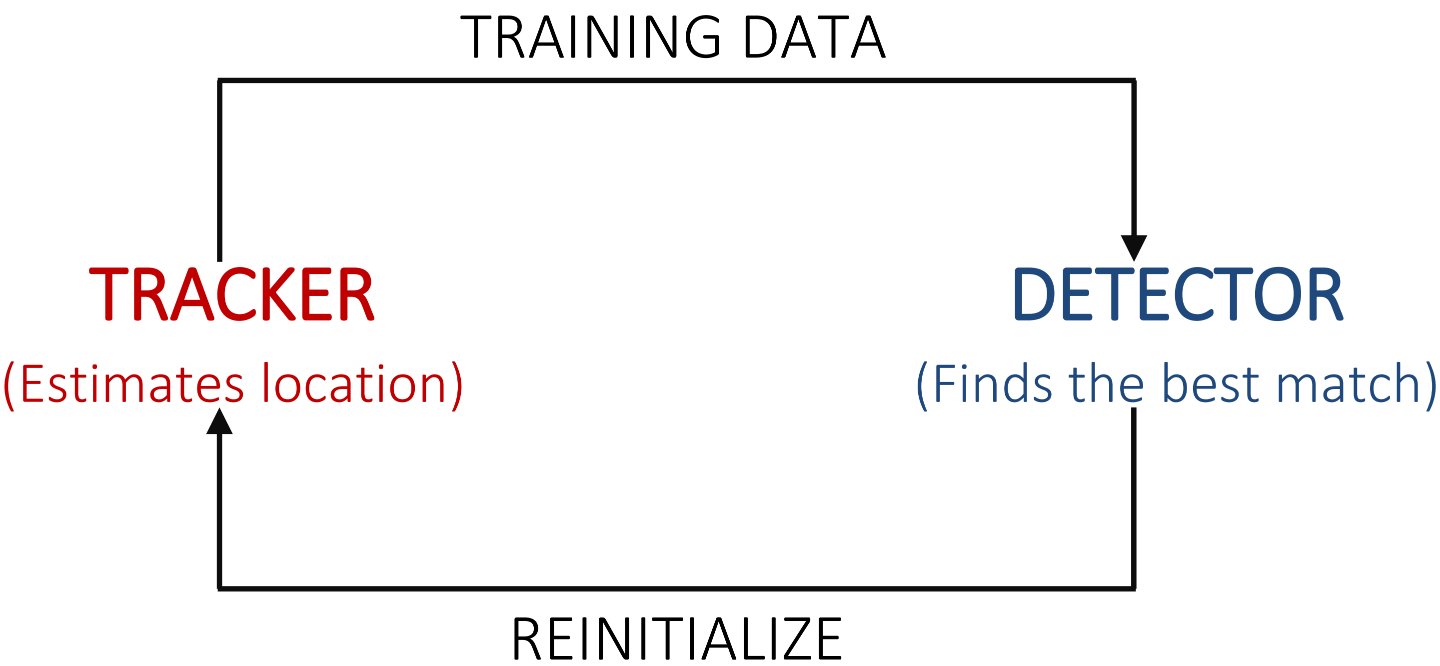
(Finds the best match)

– *Requires off-line training*
– Works over a know object
– Produces a sort of discrete trajectories
– Computationally expensive

+ *Doesn't drift*
+ *Doesn't fail if the object disappears*

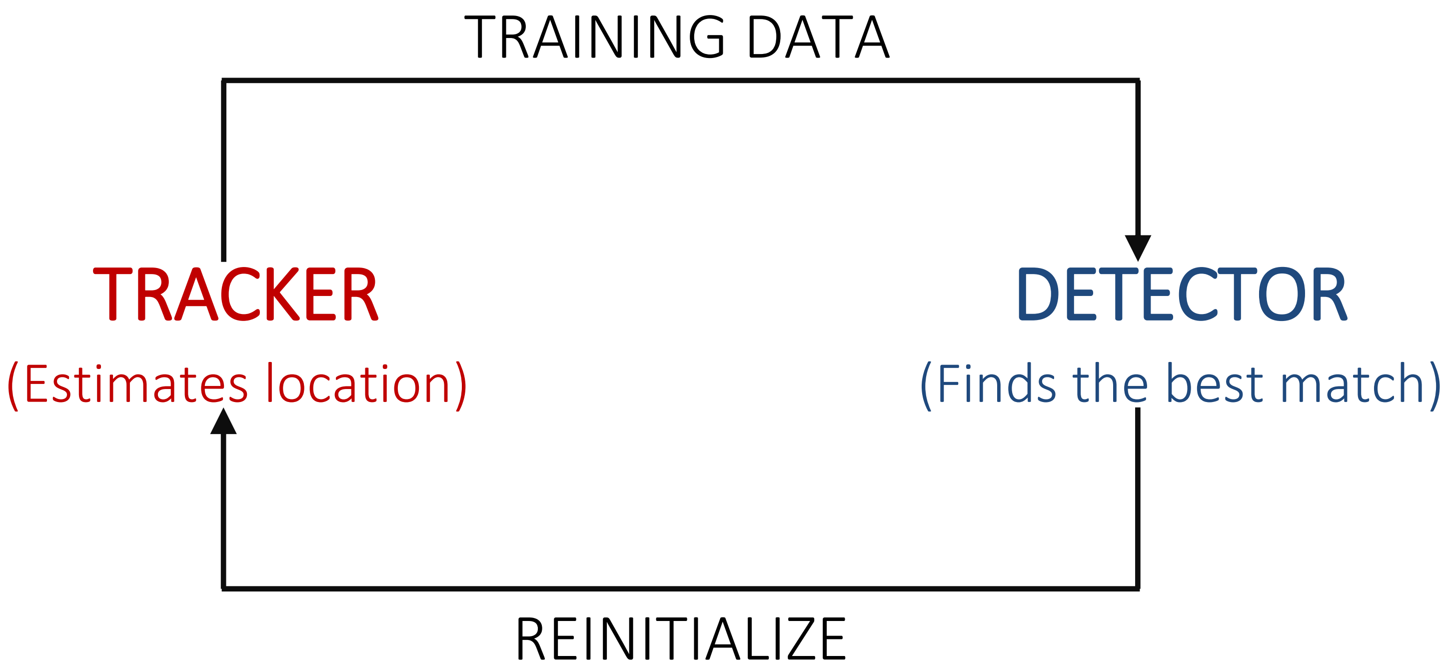
INTRODUCTION
LONG TERM TRACKING

- Interaction between tracking and detection may benefit long-term tracking.



INTRODUCTION
LONG TERM TRACKING

- Interaction between tracking and detection may benefit long-term tracking.



- How reliable is the training data?

INTRODUCTION
LONG TERM TRACKING

- Long-term tracking can be decomposed into:

TRACKING

- Tracking: follows the object from frame to frame.

INTRODUCTION
LONG TERM TRACKING

- Long-term tracking can be decomposed into:

TRACKING

DETECTION

- Tracking: follows the object from frame to frame.
- Detection: localizes the appearances that have been observed so far.

INTRODUCTION
LONG TERM TRACKING

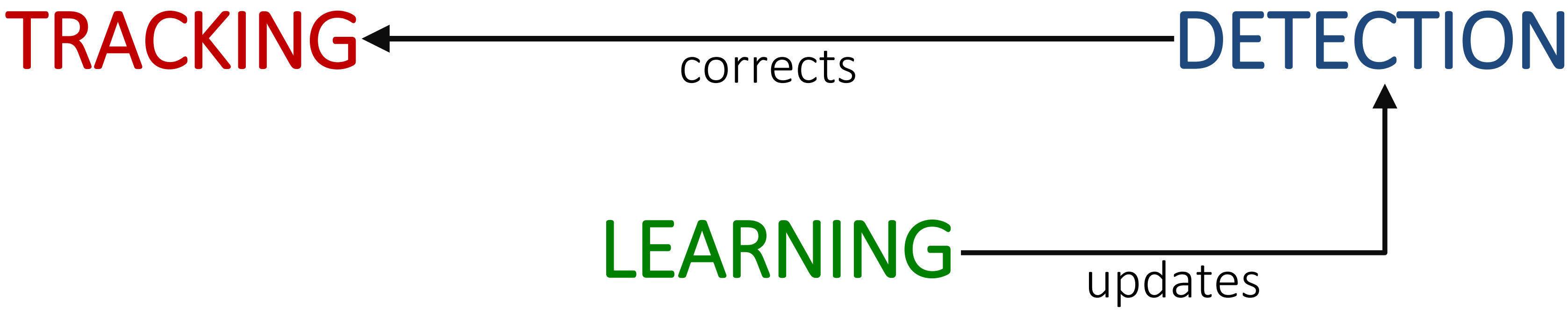
- Long-term tracking can be decomposed into:

TRACKING $\xleftarrow{\text{corrects}}$ DETECTION

- Tracking: follows the object from frame to frame.
- Detection: localizes the appearances that have been observed so far.

INTRODUCTION
LONG TERM TRACKING

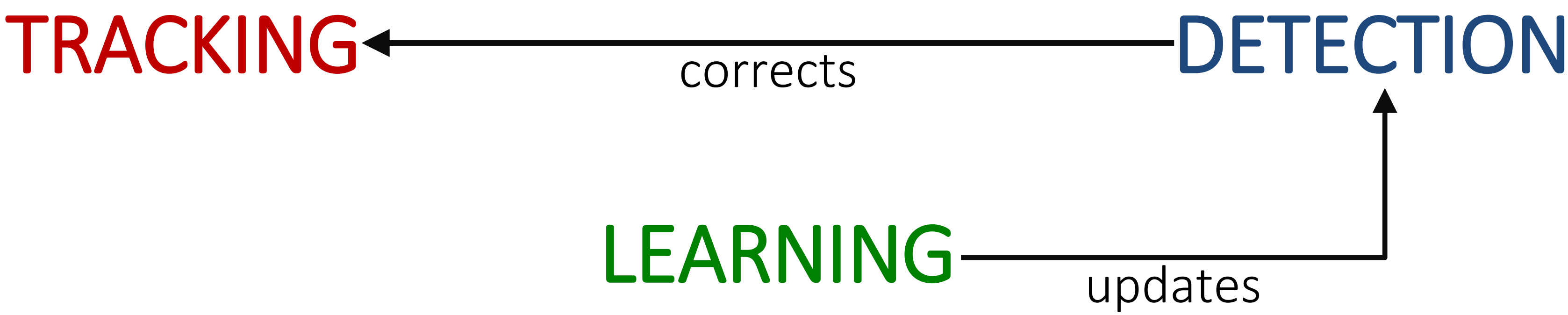
- Long-term tracking can be decomposed into:



- Tracking: follows the object from frame to frame.
- Detection: localizes the appearances that have been observed so far.
- Learning: estimates the detector errors.

INTRODUCTION
LONG TERM TRACKING

- Long-term tracking can be decomposed into:



- Tracking: follows the object from frame to frame.
- Detection: localizes the appearances that have been observed so far.
- Learning: estimates the detector errors.
  - Should deal with an arbitrary complex video stream.
  - Mustn’t degrade the detector with irrelevant information.
  - Operates in real time.

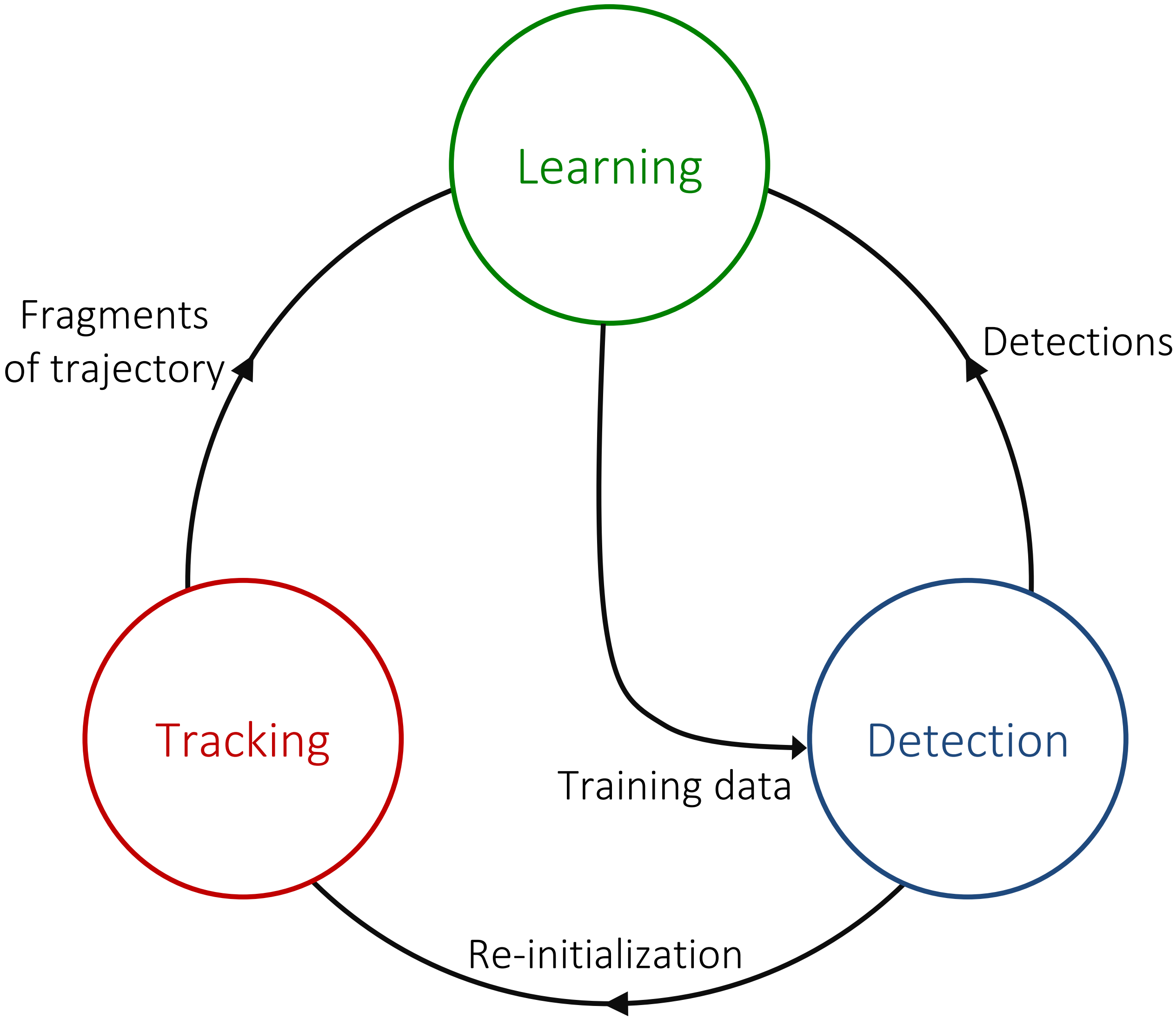
CONTENT

1. INTRODUCTION

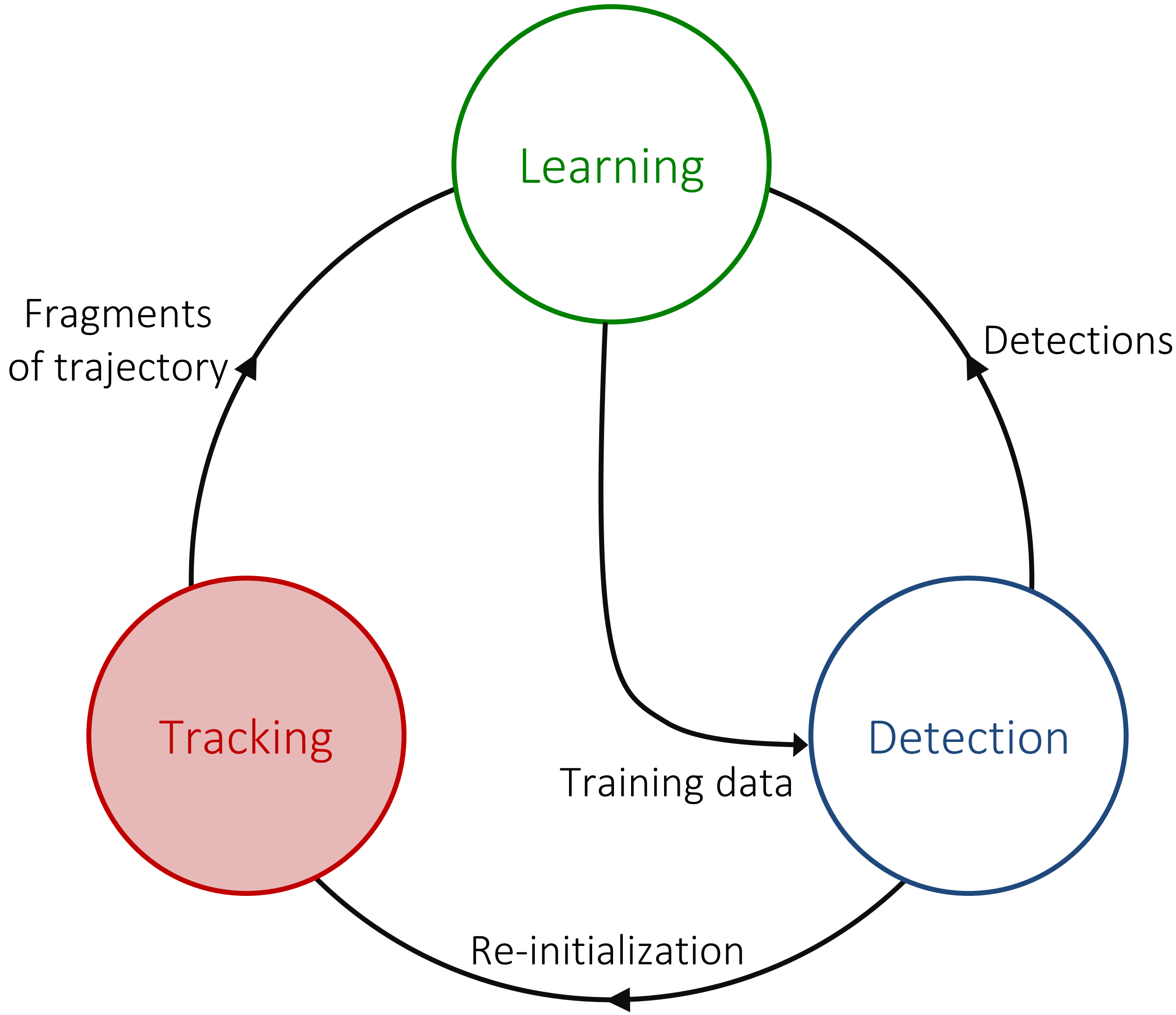
2. TLD FRAMEWORK

- TRACKING
- DETECTION
- LEARNING

TLD FRAMEWORK

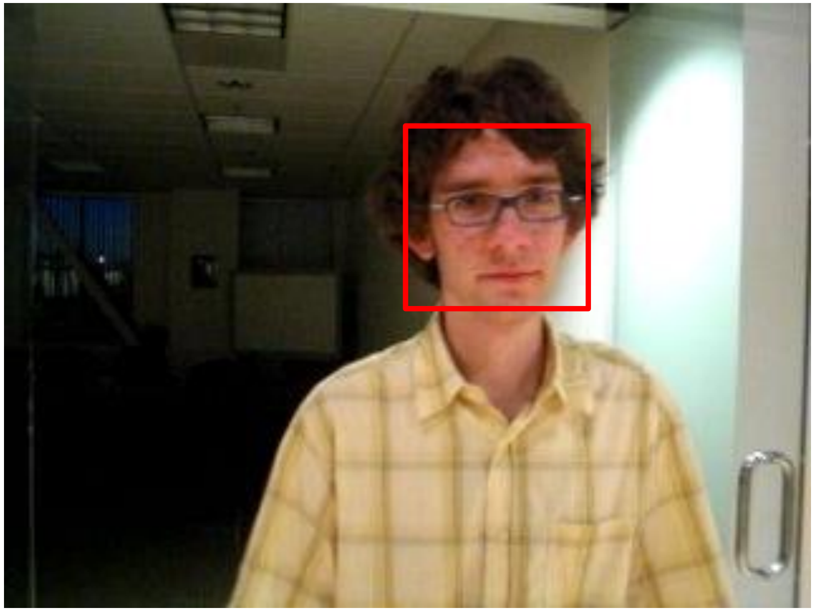


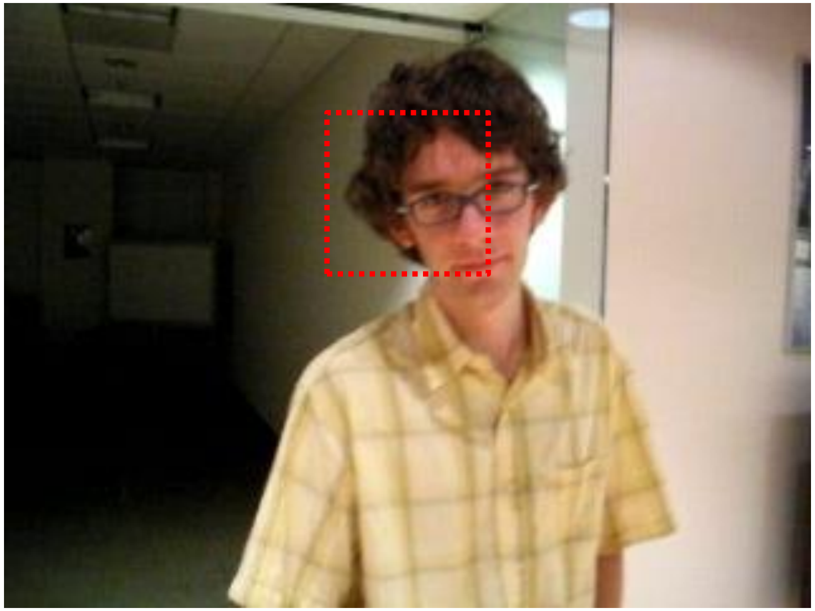
TLD FRAMEWORK



TLD FRAMEWORK
ADAPTIVE TRACKING

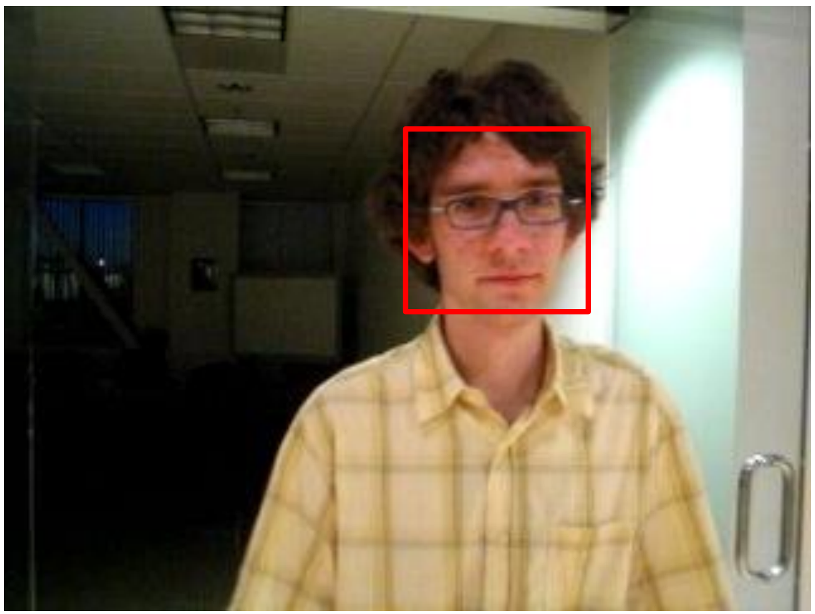
- Adaptive tracking eventually fails due to the insertion of background information into its model, commonly known as *drifting*.

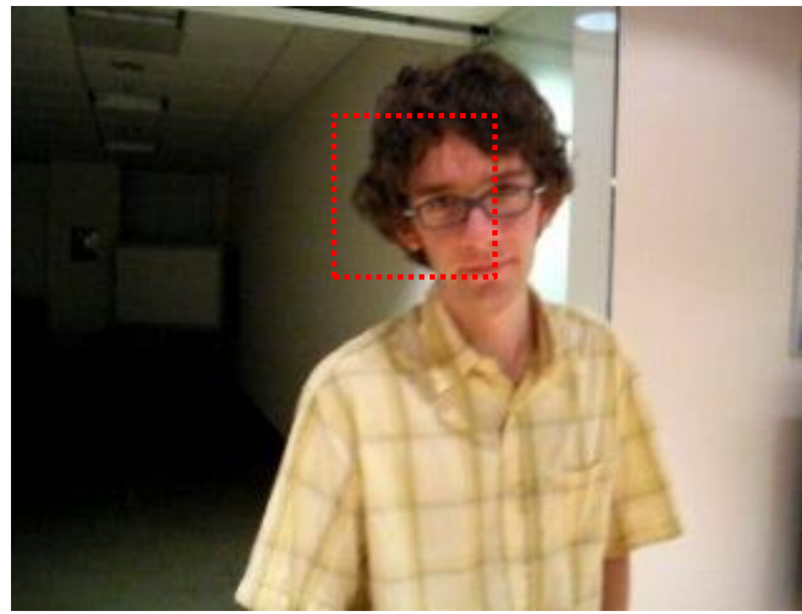




TLD FRAMEWORK
ADAPTIVE TRACKING

- Adaptive tracking eventually fails due to the insertion of background information into its model, commonly known as *drifting*.

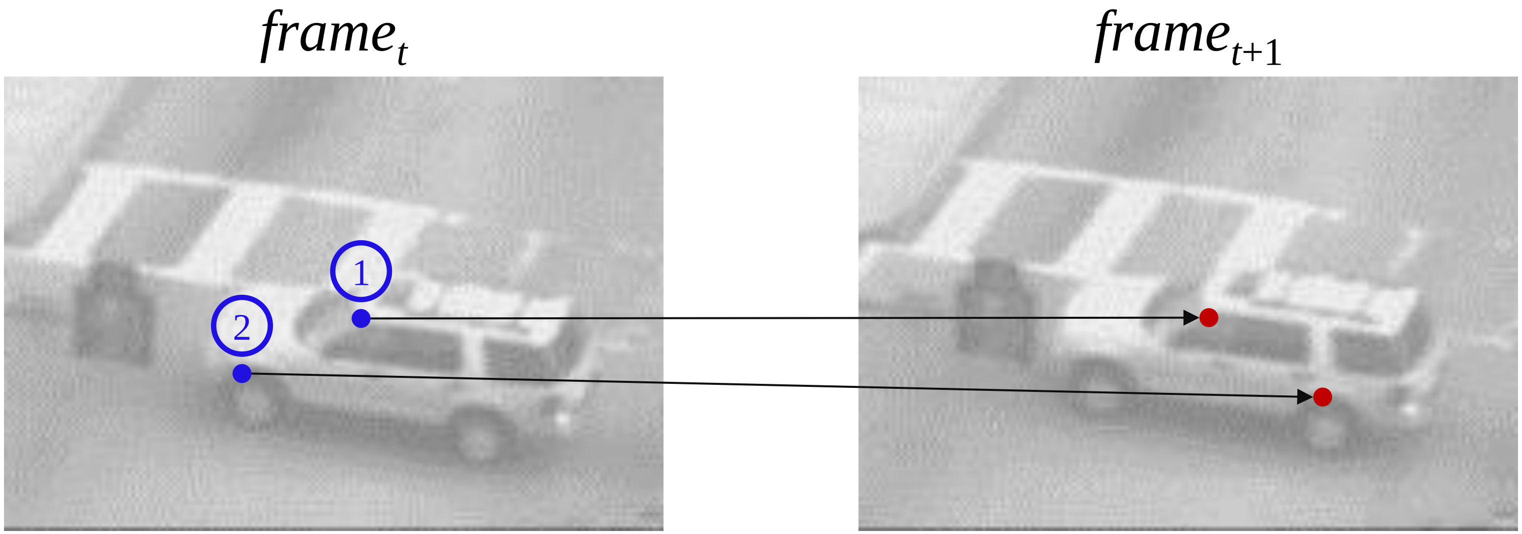




- Idea: recognize tracking failures and update only if the tracking is correct.

TLD FRAMEWORK
ADAPTIVE TRACKING

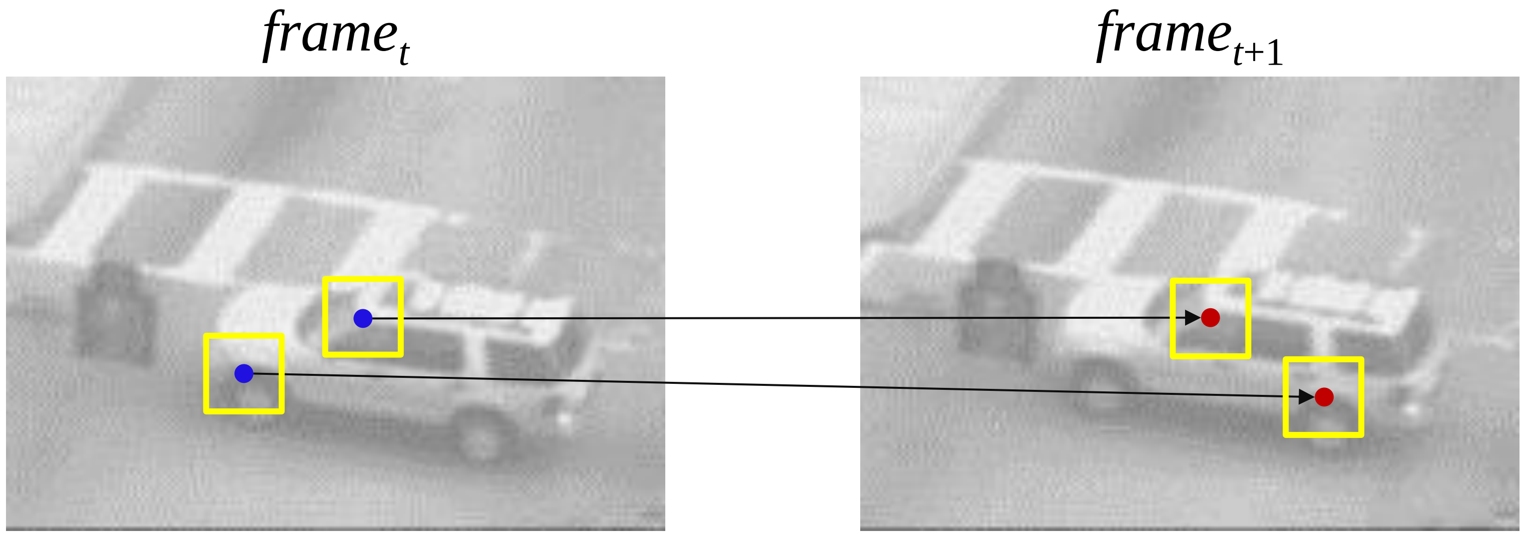
- Which of these points was tracked correctly?



Adapted from: Predator: A visual tracker that learns from its errors, Google Tech Talks.

TLD FRAMEWORK
ADAPTIVE TRACKING

- Which of these points was tracked correctly?

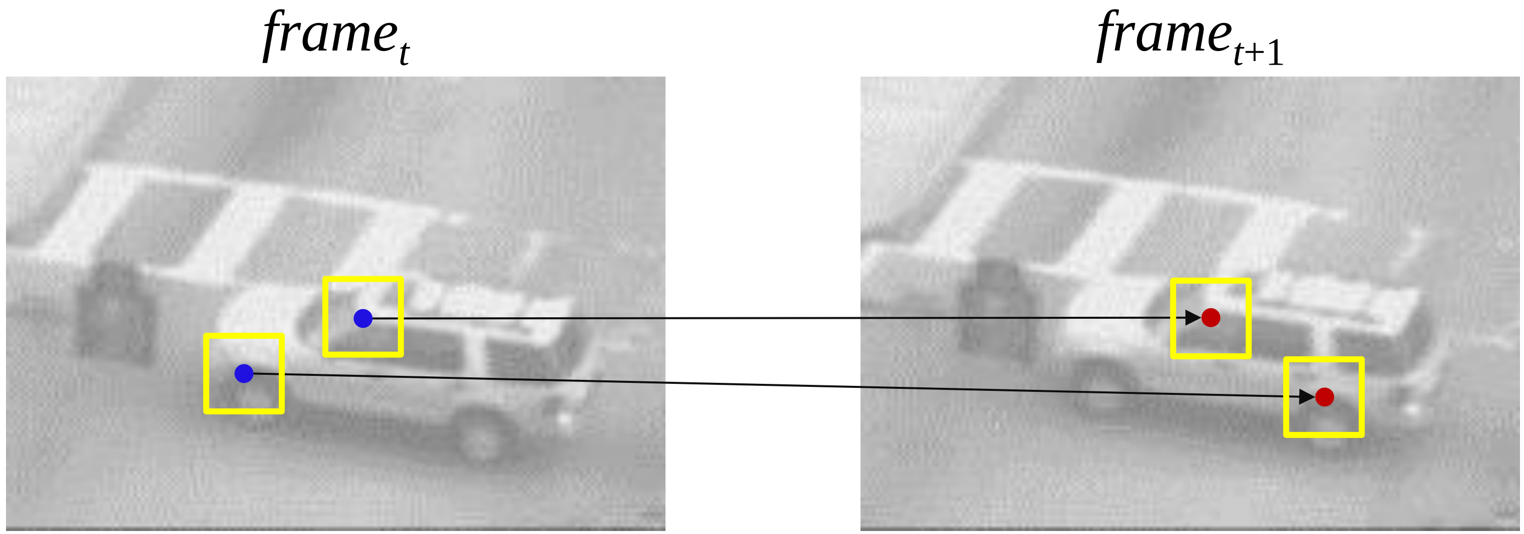


- Measure the similarity between patches around the points.

Adapted from: Predator: A visual tracker that learns from its errors, Google Tech Talks.

TLD FRAMEWORK
ADAPTIVE TRACKING

- Which of these points was tracked correctly?

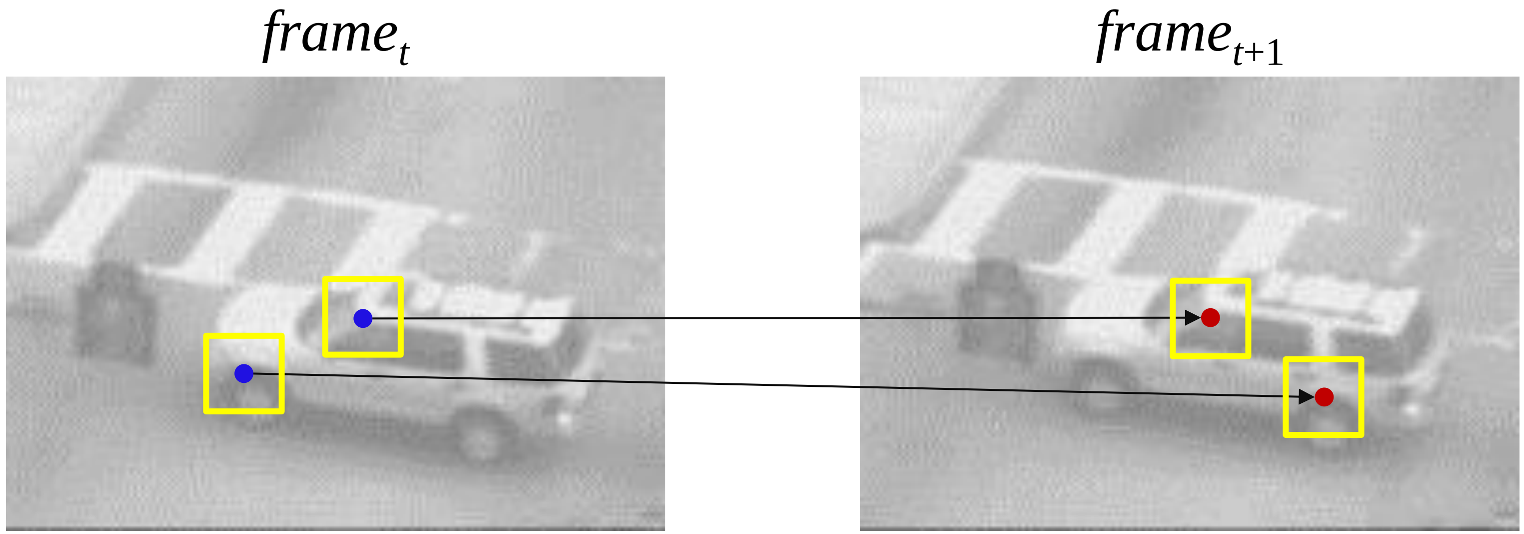


- Measure the similarity between patches around the points.
  - Typically Normalized Cross Correlation and Sum of Squared Errors between patches are embedded into tracking algorithms.

Adapted from: Predator: A visual tracker that learns from its errors, Google Tech Talks.

TLD FRAMEWORK
ADAPTIVE TRACKING

- Which of these points was tracked correctly?



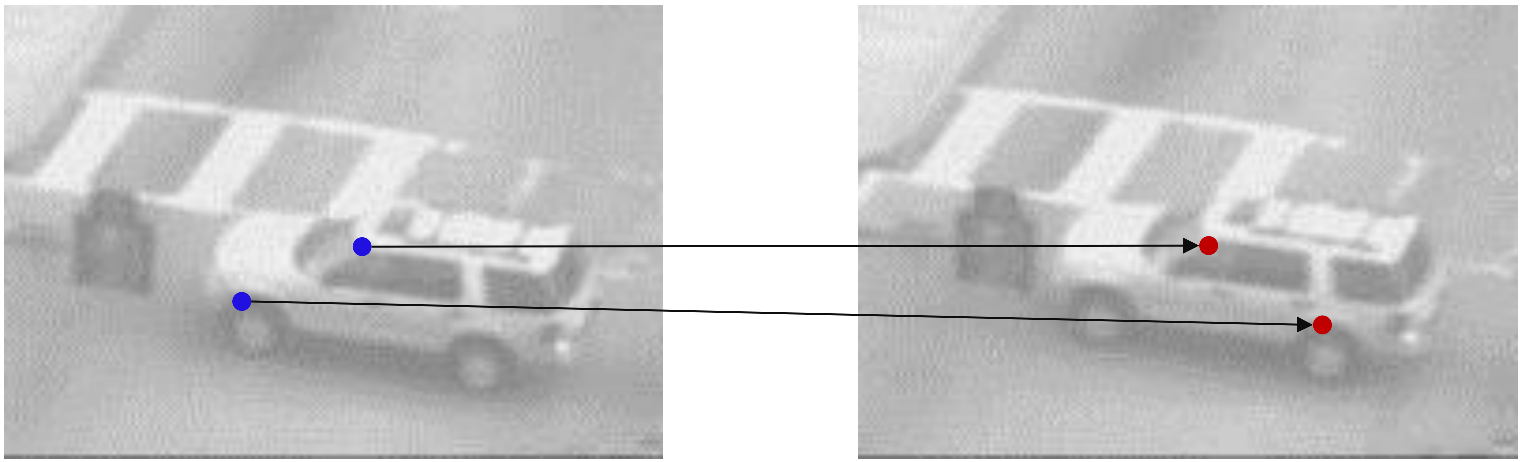
- Measure the similarity between patches around the points.
  - Typically Normalized Cross Correlation and Sum of Squared Errors between patches are embedded into tracking algorithms.
- Or, compute the Forward-Backward Error.

Adapted from: Predator: A visual tracker that learns from its errors, Google Tech Talks.

TLD FRAMEWORK
ADAPTIVE TRACKING

FORWARD-BACKWARD ERROR:

$frame_t$ $frame_{t+1}$

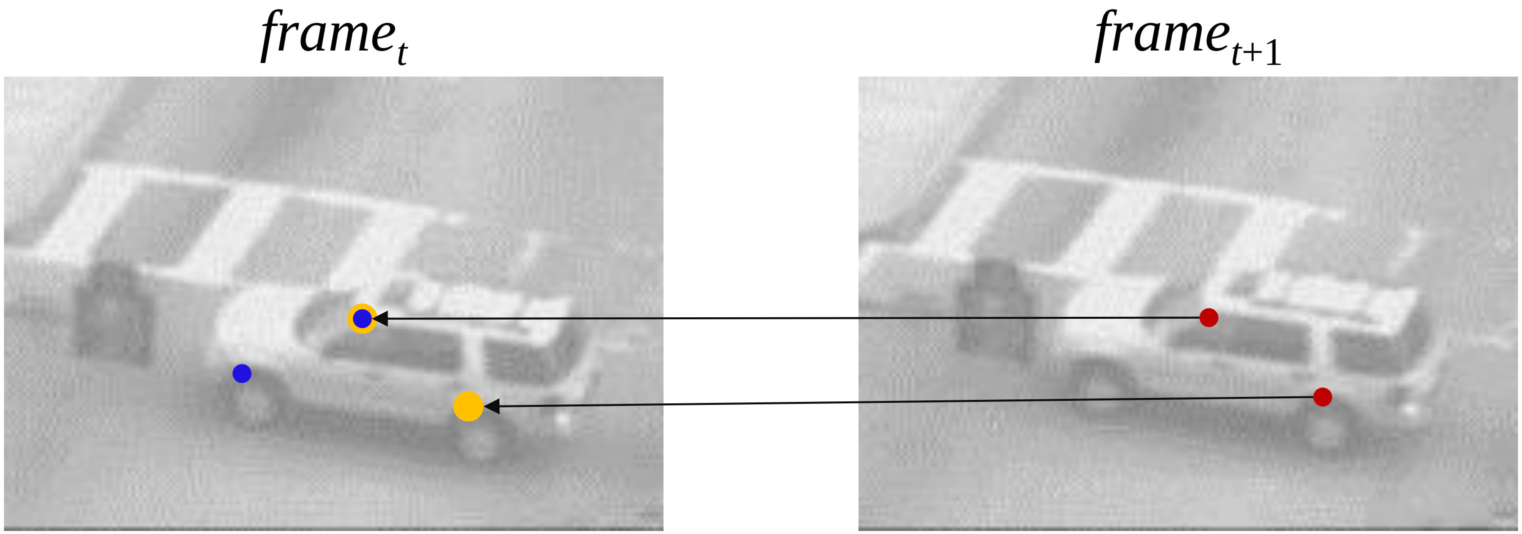


- Track points from frame t to frame $t + 1$, i.e., Forward tracking.

Adapted from: Predator: A visual tracker that learns from its errors, Google Tech Talks.

TLD FRAMEWORK
ADAPTIVE TRACKING

FORWARD-BACKWARD ERROR:

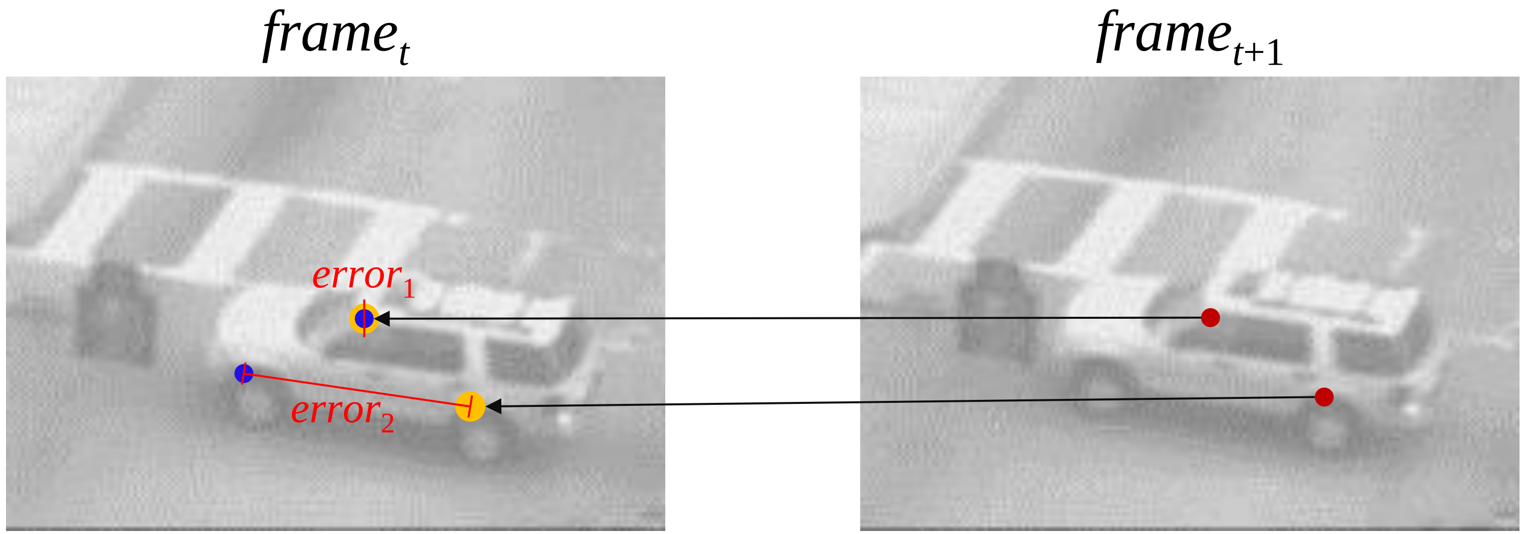


- Track points from frame t to frame $t + 1$, i.e., Forward tracking.
- Track points from frame $t + 1$ to frame t, i.e., Backward tracking.

Adapted from: Predator: A visual tracker that learns from its errors, Google Tech Talks.

TLD FRAMEWORK
ADAPTIVE TRACKING

FORWARD-BACKWARD ERROR:



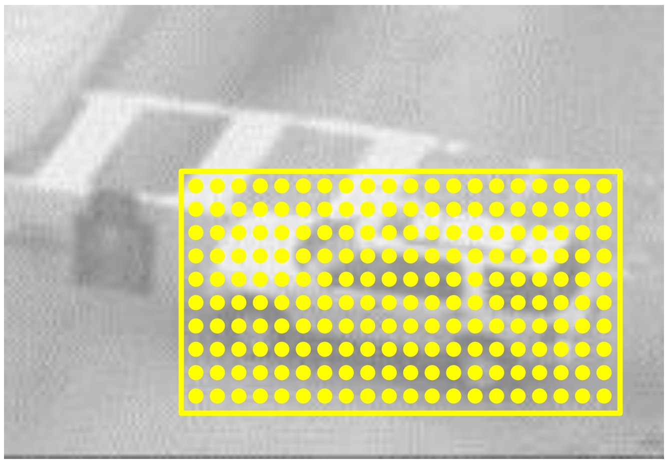
- Track points from frame t to frame $t + 1$, i.e., Forward tracking.
- Track points from frame $t + 1$ to frame t, i.e., Backward tracking.
- Compute the error between initial points and backward points.
  - Small errors = correctly tracked points.

Adapted from: Predator: A visual tracker that learns from its errors, Google Tech Talks.

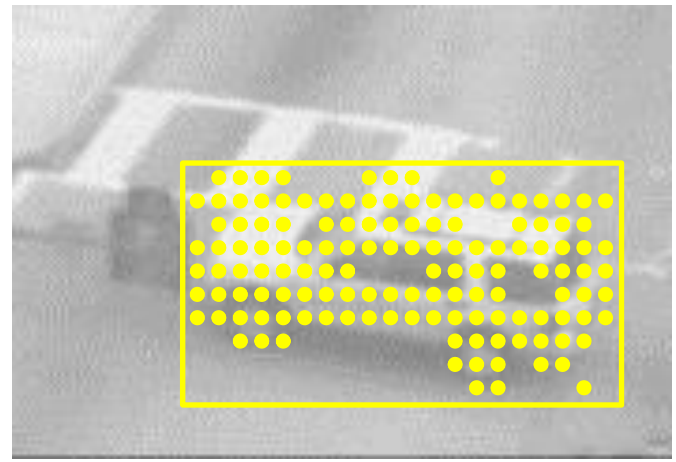
TLD FRAMEWORK
ADAPTIVE TRACKING

MEDIAN FLOW TRACKER

$frame_t$



$frame_{t+1}$



Initialize a grid

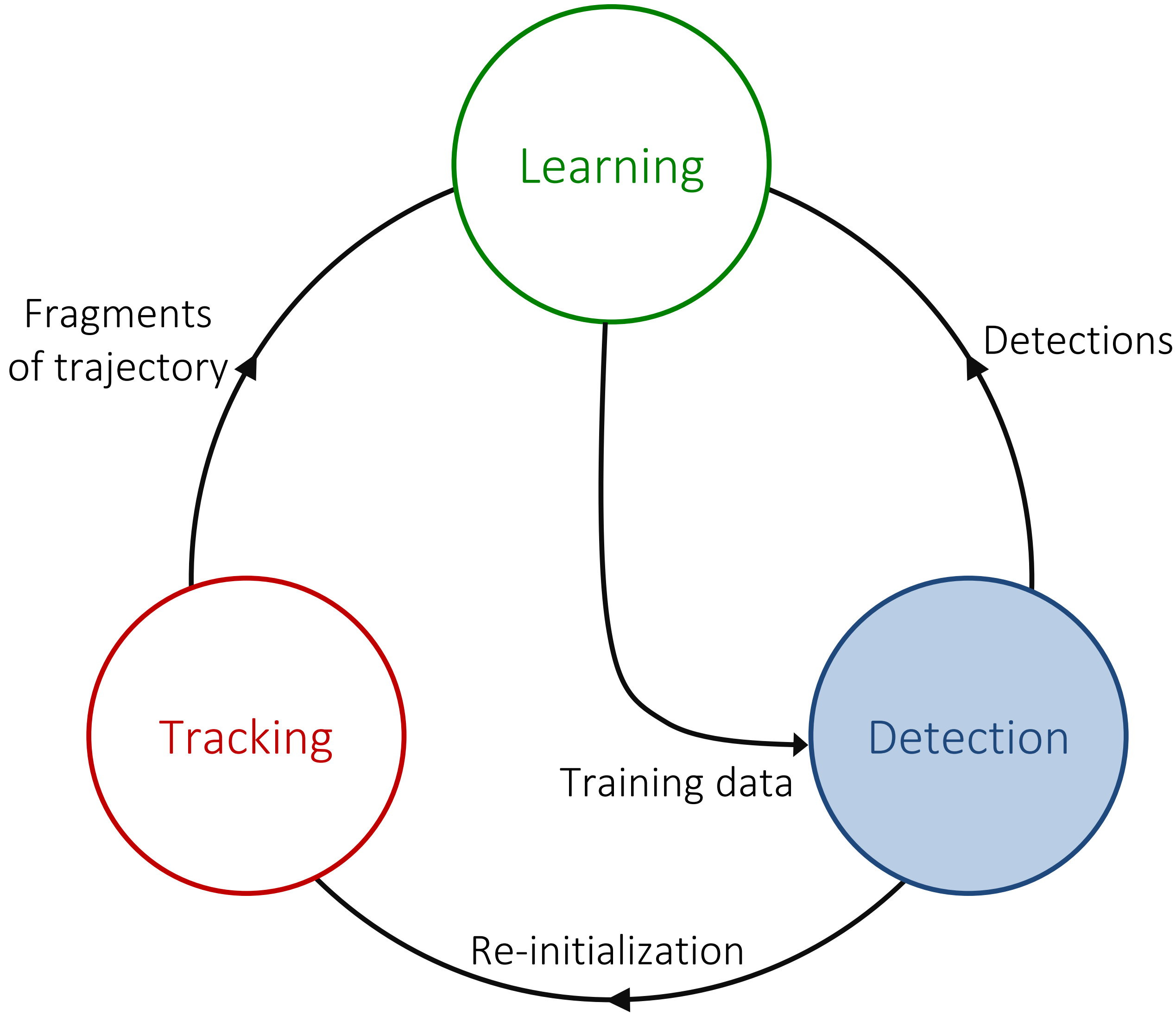
Track points between frames

Estimate point reliability

Filter out 50% outliers

Estimate Bounding box

TLD FRAMEWORK



TLD FRAMEWORK
DETECTION

- The goal is to discriminate the object from the background, i.e., model the appearance of the object.

- Let M be the object model, so that:

$$M = \left\{p_1^+, p_2^+, \ldots, p_m^+, p_1^-, p_2^-, \ldots, p_n^-\right\}$$

Where,

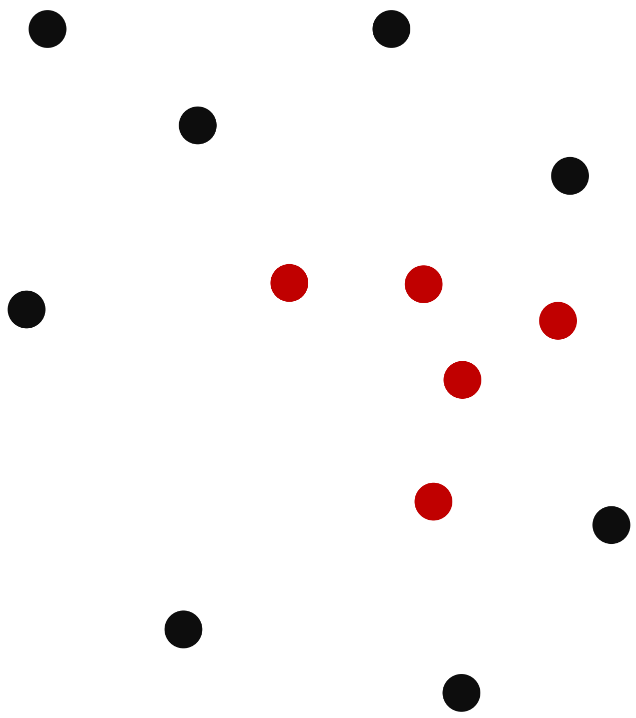
– p^+, represents the object patches.

– p^-, are the background patches.

- New image patches can be classified as belonging to the object or the background using a Nearest Neighbor Classifier (NNC).

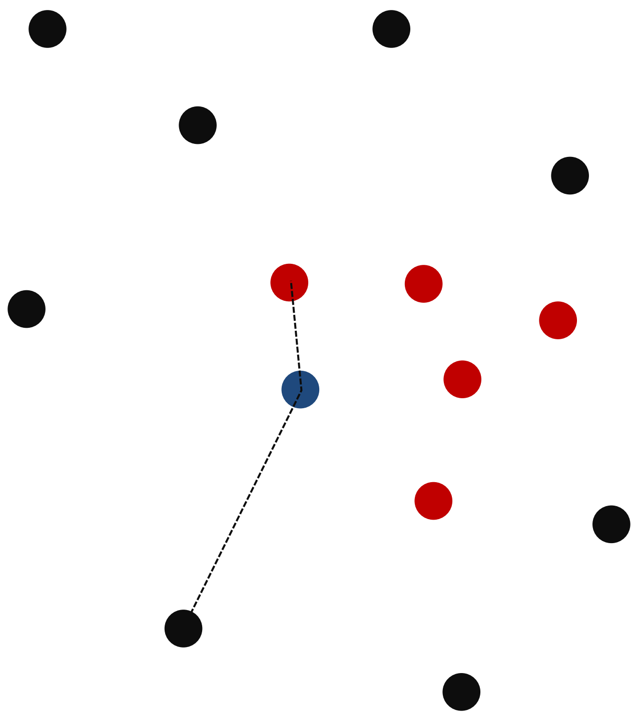
TLD FRAMEWORK
DETECTION

NEAREST NEIGHBOR CLASSIFIER



- Red and black points represent the object and background patches, respectively, in a d-dimensional space.

NEAREST NEIGHBOR CLASSIFIER



- Use a relative similarity, S^r, to classify a new patch (blue point), formally:

$$S^r = \frac{S^+(p,M)}{S^+(p,M) + S^-(p,M)}$$

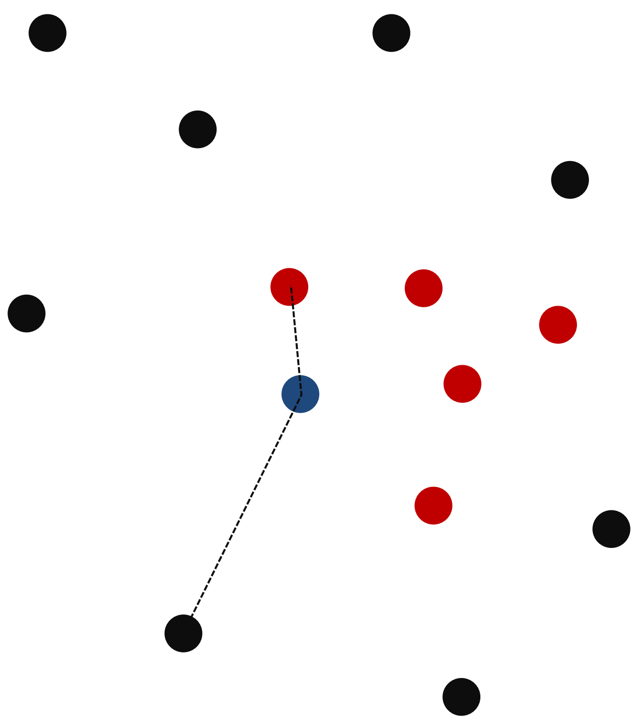
Where,

– S^+, is the similarity with the positive nearest neighbor.

– S^-, is the similarity with the negative nearest neighbor.

- Red and black points represent the object and background patches, respectively, in a d-dimensional space.

NEAREST NEIGHBOR CLASSIFIER



- Similarity between patches, $S(p_i, p_j)$, is given by:

$$S(p_i, p_j) = 0.5\left(\mathrm{NCC}(p_i, p_j) + 1\right)$$

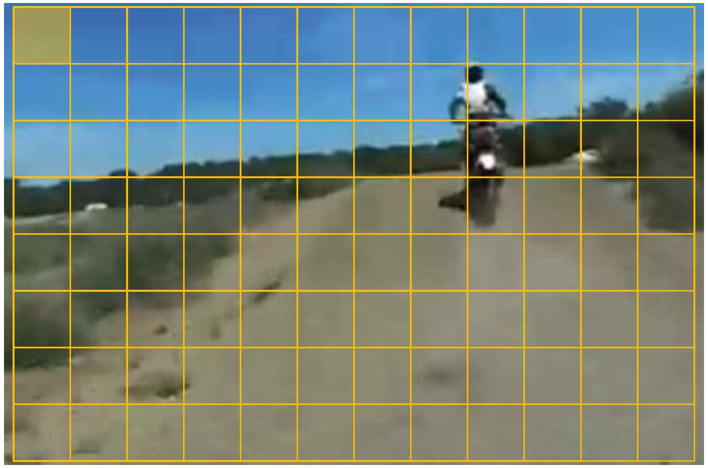
Where,

  – NCC, is the Normalized Cross Correlation between patches p_i and p_j.

- Red and black points represent the object and background patches, respectively, in a d-dimensional space.

TLD FRAMEWORK
DETECTION

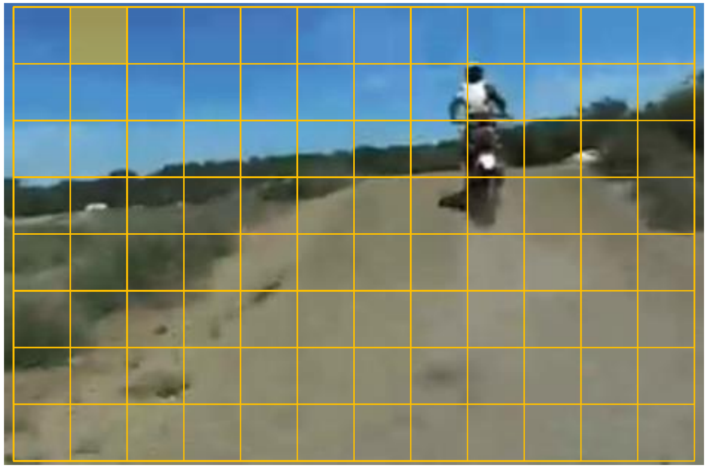
- Image patches are generated from the initial bounding box, for example, a QVGA image (240x320) with the following parameters:
  - Scale step = 1.2,
  - Horizontal step = 0.1(object's width),
  - Vertical step = 0.1(object's height),
  - Minimal bounding box size = 20,



Produces around 50K bounding boxes.

TLD FRAMEWORK
DETECTION

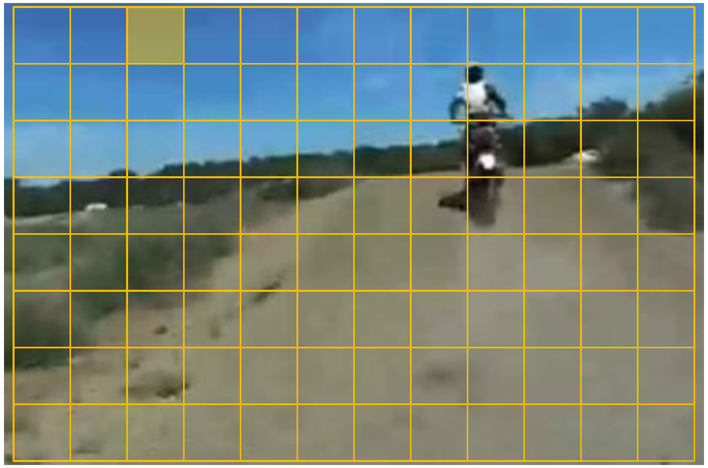
- Image patches are generated from the initial bounding box, for example, a QVGA image (240x320) with the following parameters:
  - Scale step = 1.2,
  - Horizontal step = 0.1(object's width),
  - Vertical step = 0.1(object's height),
  - Minimal bounding box size = 20,



Produces around 50K bounding boxes.

TLD FRAMEWORK
DETECTION

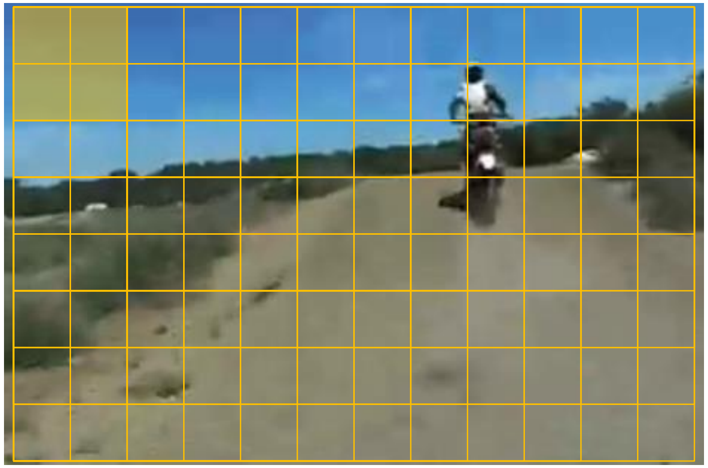
- Image patches are generated from the initial bounding box, for example, a QVGA image (240x320) with the following parameters:
  - Scale step = 1.2,
  - Horizontal step = 0.1(object's width),
  - Vertical step = 0.1(object's height),
  - Minimal bounding box size = 20,



Produces around 50K bounding boxes.

TLD FRAMEWORK
DETECTION

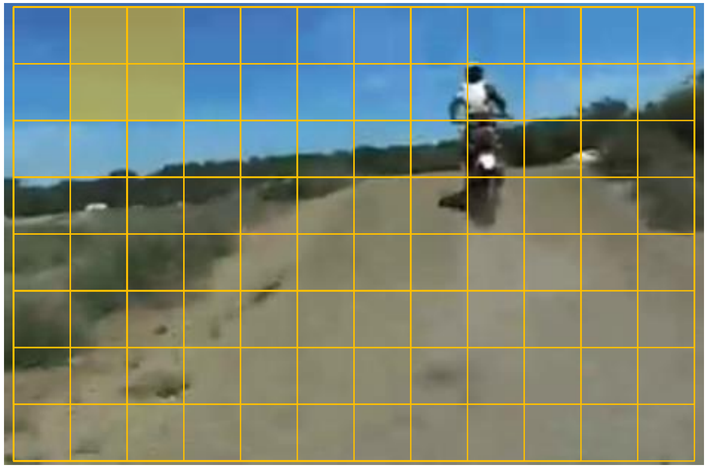
- Image patches are generated from the initial bounding box, for example, a QVGA image (240x320) with the following parameters:
  - Scale step = 1.2,
  - Horizontal step = 0.1(object's width),
  - Vertical step = 0.1(object's height),
  - Minimal bounding box size = 20,



Produces around 50K bounding boxes.

TLD FRAMEWORK
DETECTION

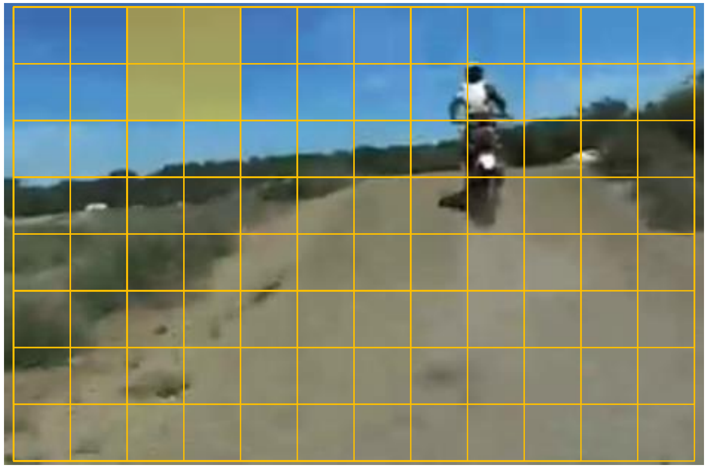
- Image patches are generated from the initial bounding box, for example, a QVGA image (240x320) with the following parameters:
  - Scale step = 1.2,
  - Horizontal step = 0.1(object's width),
  - Vertical step = 0.1(object's height),
  - Minimal bounding box size = 20,



Produces around 50K bounding boxes.

TLD FRAMEWORK
DETECTION

- Image patches are generated from the initial bounding box, for example, a QVGA image (240x320) with the following parameters:
  - Scale step = 1.2,
  - Horizontal step = 0.1(object's width),
  - Vertical step = 0.1(object's height),
  - Minimal bounding box size = 20,

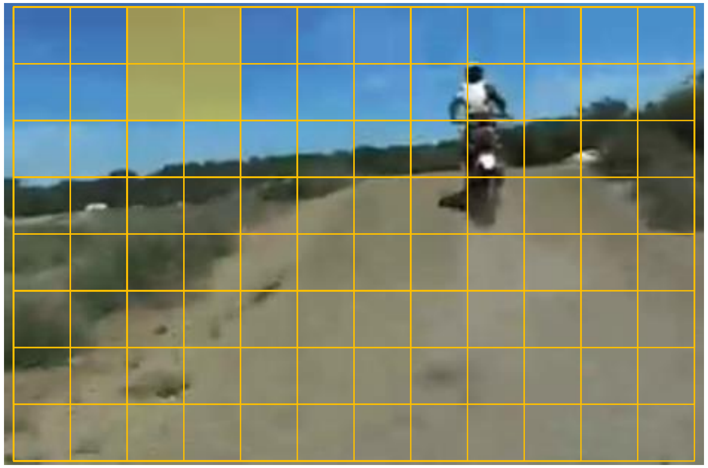


Produces around 50K bounding boxes.

TLD FRAMEWORK
DETECTION

- Image patches are generated from the initial bounding box, for example, a QVGA image (240x320) with the following parameters:

  – Scale step = 1.2,
  – Horizontal step = 0.1(object's width),
  – Vertical step = 0.1(object's height),
  – Minimal bounding box size = 20,



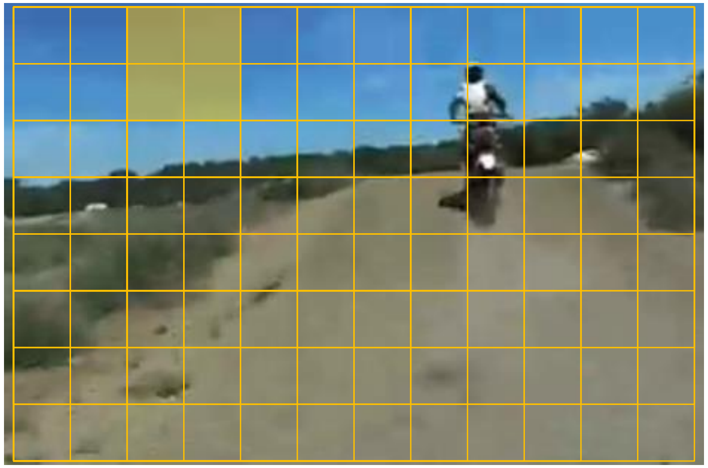
  Produces around 50K bounding boxes.

- Evaluate the NNC over every possible patch in the image is an unfeasible task as it involves evaluation of the relative similarity.

TLD FRAMEWORK
DETECTION

- Image patches are generated from the initial bounding box, for example, a QVGA image (240x320) with the following parameters:

  - Scale step = 1.2,
  - Horizontal step = 0.1(object's width),
  - Vertical step = 0.1(object's height),
  - Minimal bounding box size = 20,



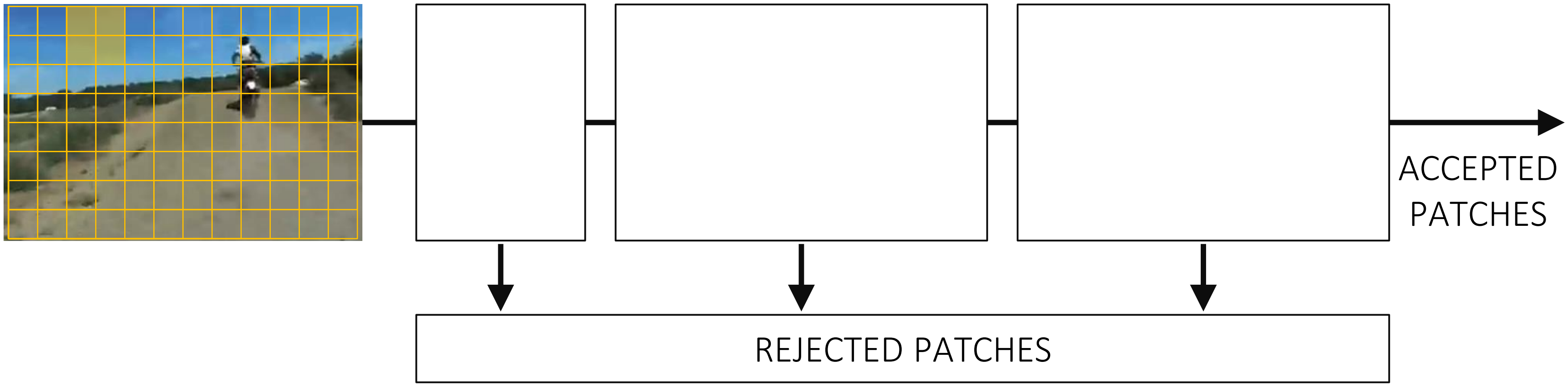
  Produces around 50K bounding boxes.

- Evaluate the NNC over every possible patch in the image is an unfeasible task as it involves evaluation of the relative similarity.

USE A CASCADE OF CLASSIFIERS INSTEAD!

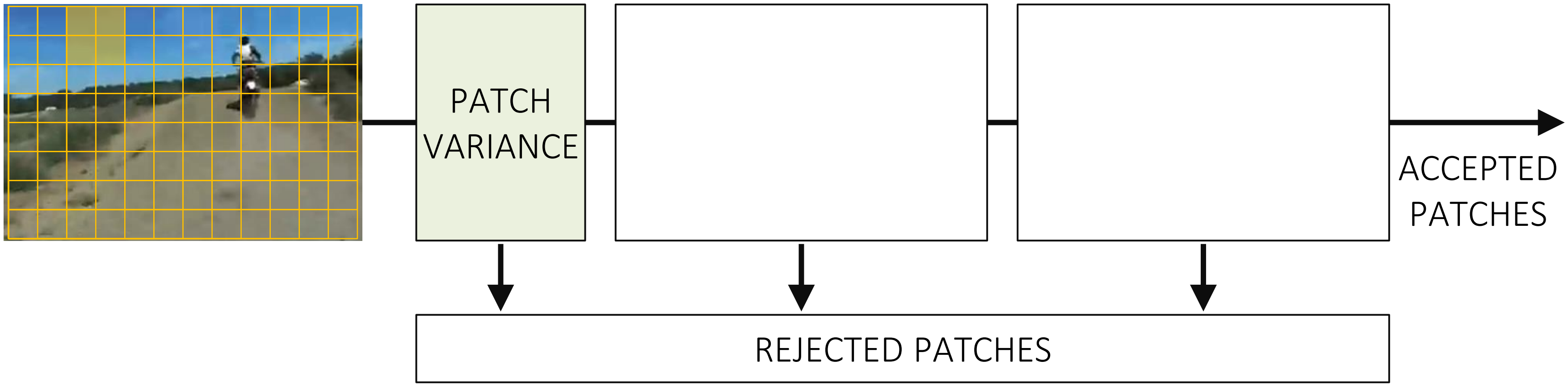
TLD FRAMEWORK
DETECTION

CASCADE OF CLASSIFIERS



TLD FRAMEWORK
DETECTION

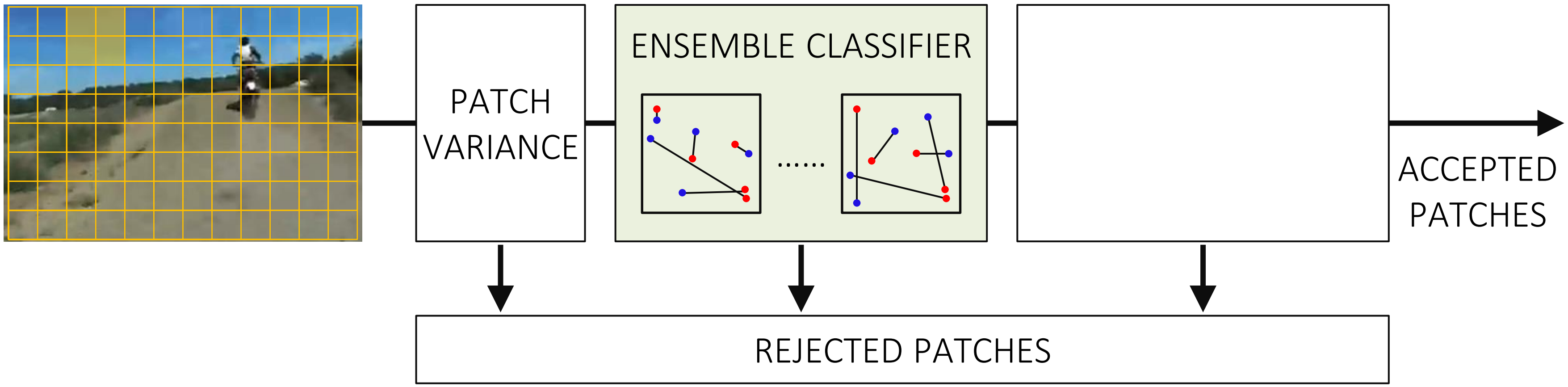
CASCADE OF CLASSIFIERS



- Rejects patches that have variance smaller than 50% of the initial bounding box variance.

TLD FRAMEWORK
DETECTION

CASCADE OF CLASSIFIERS



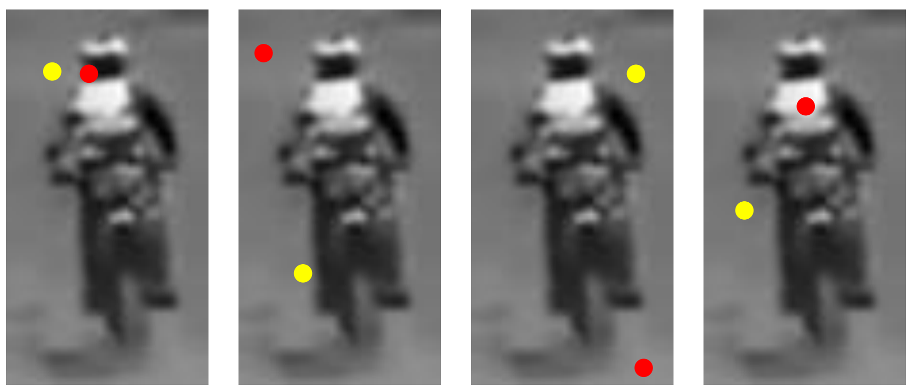
- An Ensemble Classifier consist of n base classifiers.
- Each base classifier, c_i, performs a number of pixel comparisons, f_k, for each patch, resulting in a binary code, x_i.

TLD FRAMEWORK
DETECTION

CASCADE OF CLASSIFIERS

- Consider an image patch, p, and an Ensemble Classifier which has a single base classifier to perform four pixel comparisons, f_k, so that:

$$f_k = \begin{cases} 1, & p(\text{position}_{k,red}) > p(\text{position}_{k,yellow}) \\ 0, & \text{otherwise} \end{cases}$$

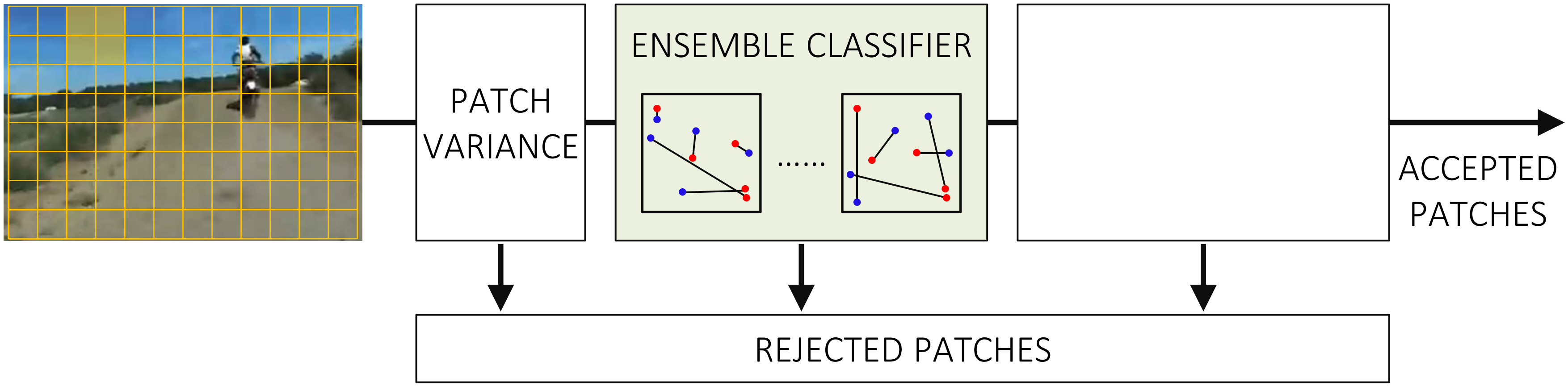


$f_1 = 0 \qquad f_2 = 1 \qquad f_3 = 0 \qquad f_4 = 1$

- Finally, the binary code x is equal to 5.

TLD FRAMEWORK
DETECTION

CASCADE OF CLASSIFIERS



- An Ensemble Classifier consist of n base classifiers.
- Each base classifier, c_i, performs a number of pixel comparisons, f_k, for each patch, resulting in a binary code, x_i.
- An image patch is classified as "*object*" if the mean of the posterior probabilities:

$$\overline{P} = \frac{1}{n} \sum_{i=1}^{n} P_i\left(y \middle| x_i\right)$$

is greater than 0.5.

CASCADE OF CLASSIFIERS

- The posterior probability for the *i-th* base classifier is given by:

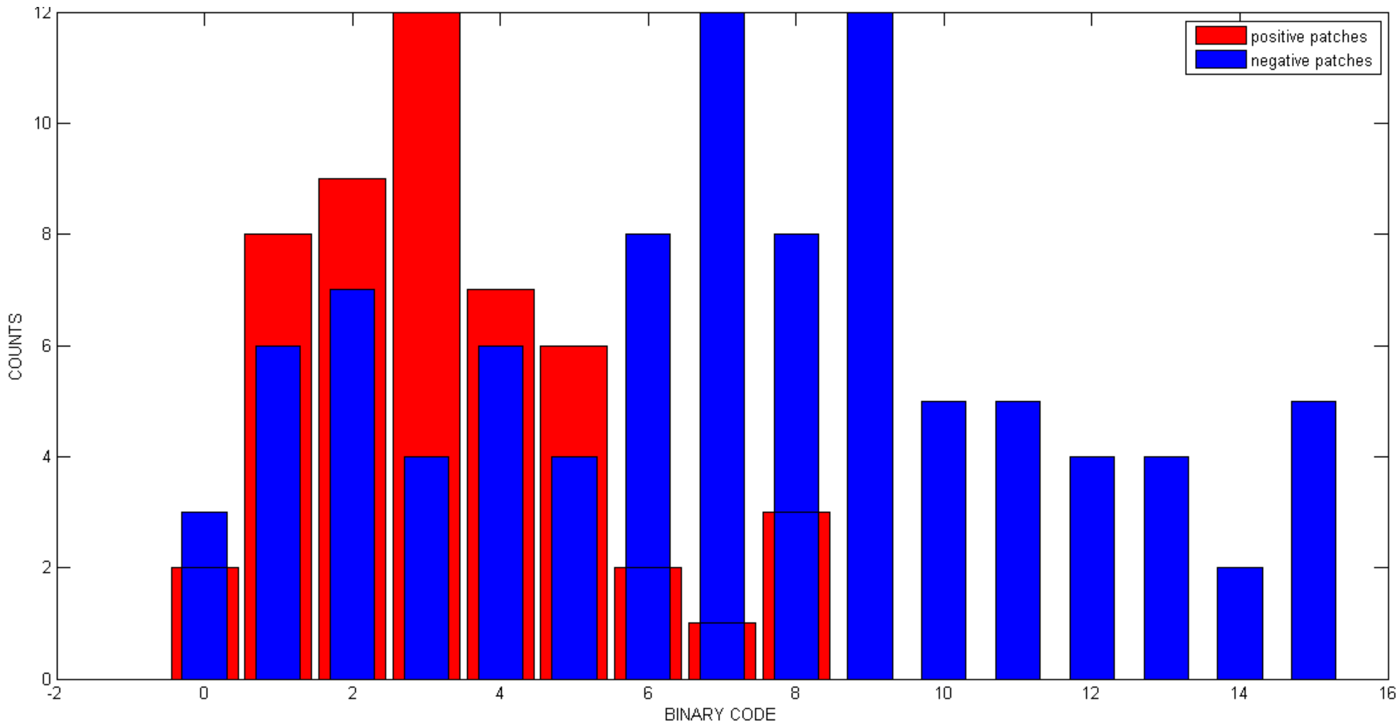
$$P_i(y|x_i) = \frac{n^+(x_i)}{n^+(x_i) + n^-(x_i)}$$

Where, $n^+(x_i)$ and $n^-(x_i)$ correspond to the number of positive and negative patches, respectively, that were assigned the same binary code.

TLD FRAMEWORK
DETECTION

CASCADE OF CLASSIFIERS

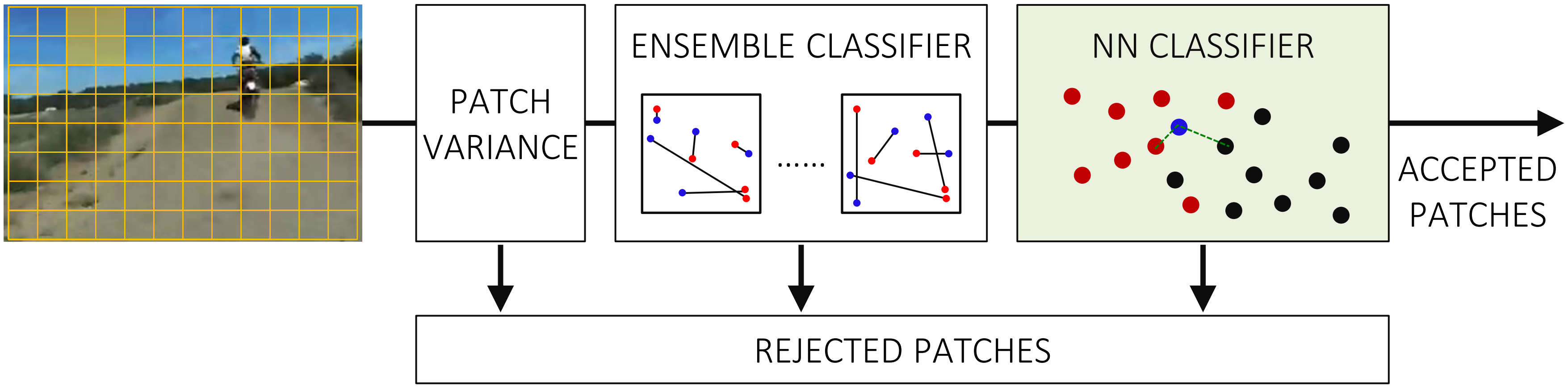
- For example, consider a trained base classifier and a $x_i = 5$, then:



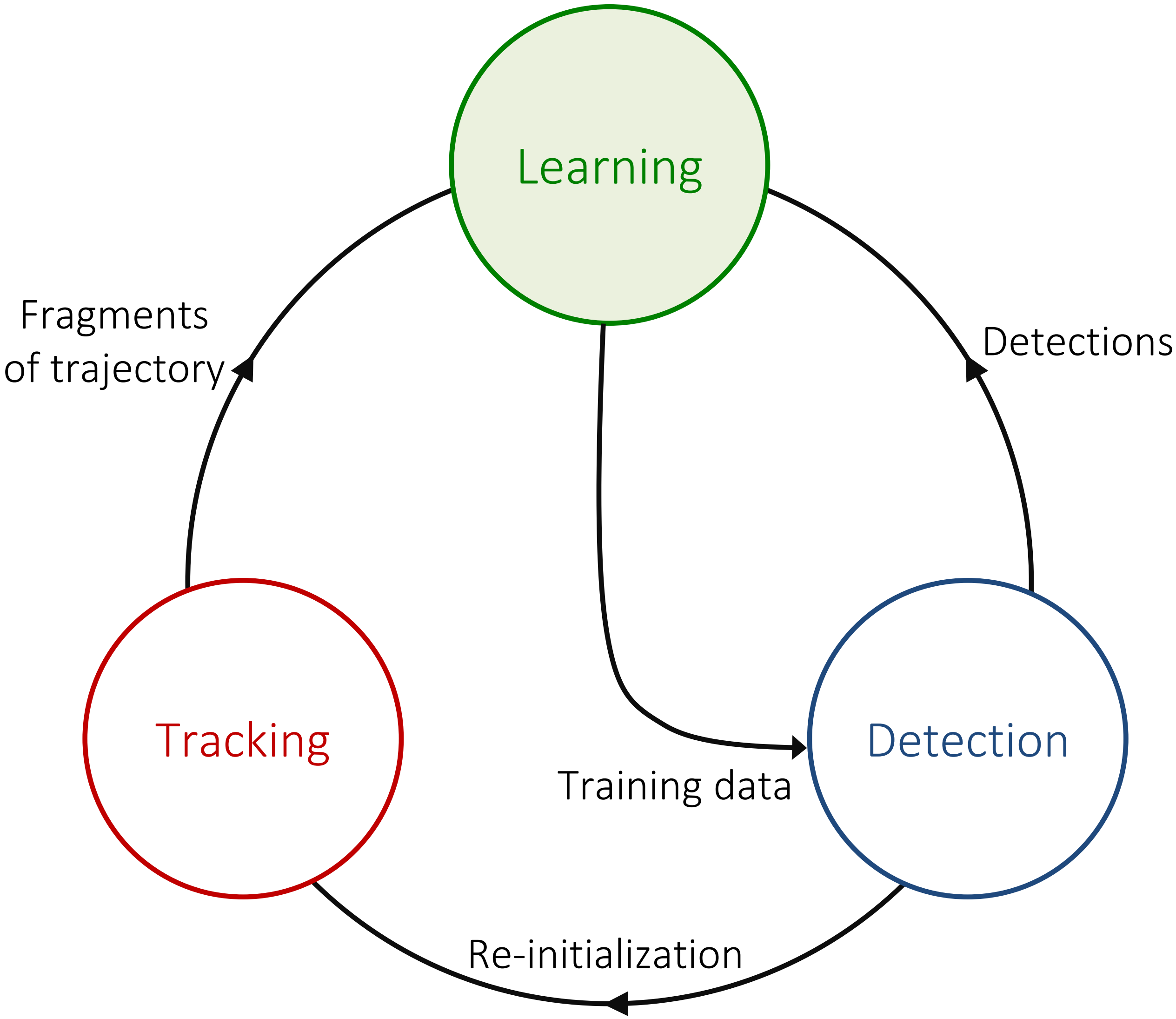
$$P_1(y|x_1 = 5) = \frac{6}{6+4} = \frac{6}{10} = 0.6$$

TLD FRAMEWORK
DETECTION

CASCADE OF CLASSIFIERS



TLD FRAMEWORK



TLD FRAMEWORK
LEARNING

- Use online learning techniques to improve the performance of the detector.

- Online learning is challenging…
  - Lack of training data
  - Unlabeled data
  - Requires real time processing.

- Assuming that these challenges are overcome…
  - Identify the errors committed by the detector in order to update it and avoid these errors in the future.

TLD FRAMEWORK LEARNING

- Use online learning techniques to improve the performance of the detector.

- Online learning is challenging…
  - Lack of training data
  - Unlabeled data
  - Requires real time processing.

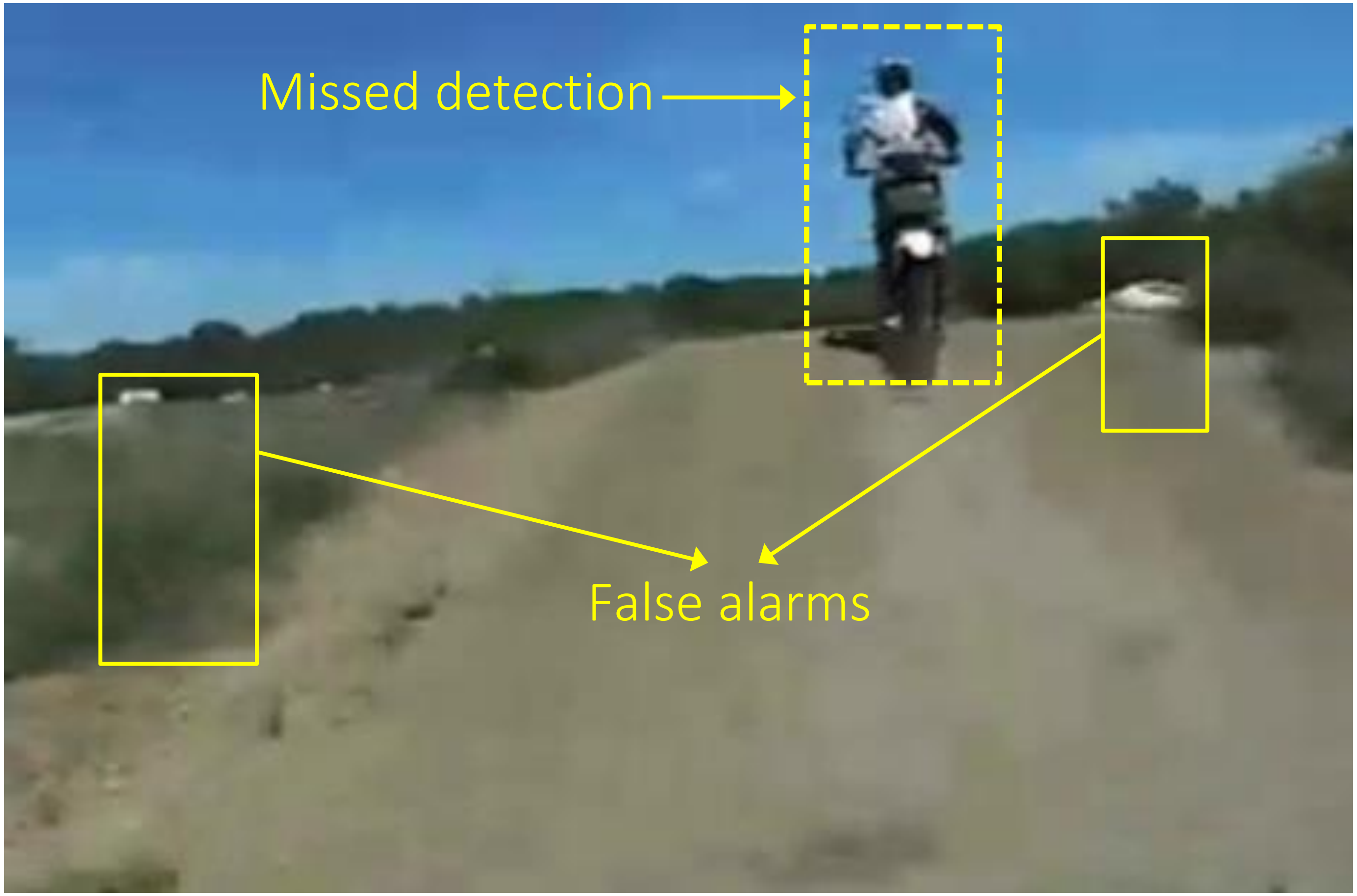
- Assuming that these challenges are overcome…
  - Identify the errors committed by the detector in order to update it and avoid these errors in the future.

P-experts

N-experts

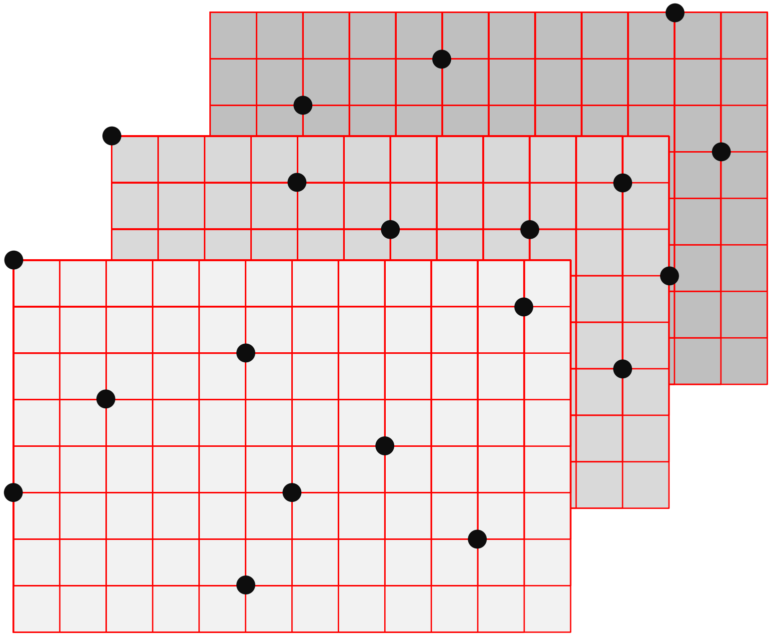
TLD FRAMEWORK
LEARNING

– Identify the errors committed by the detector in order to update it and avoid these errors in the future.



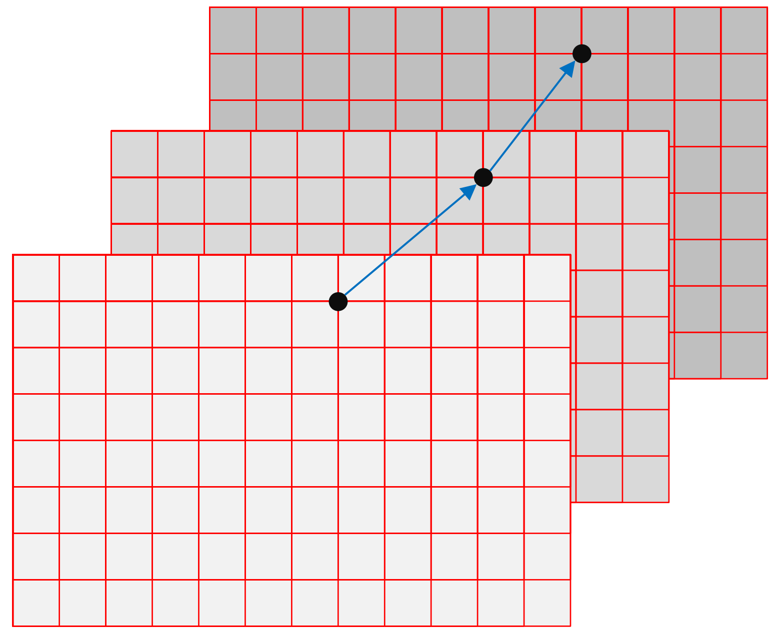
TLD FRAMEWORK LEARNING

– N-expert explores the **spatial structure** in the data in order to identify false positives (false alarms).

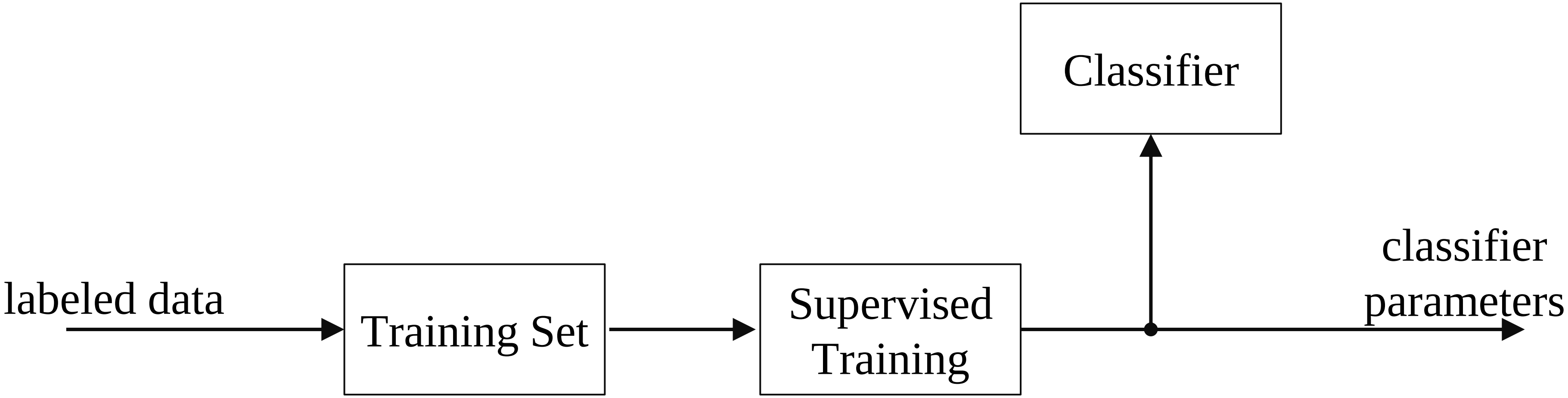


TLD FRAMEWORK LEARNING

– P-expert explores the **temporal structure** in the data in order to identify false negatives (missed detections).

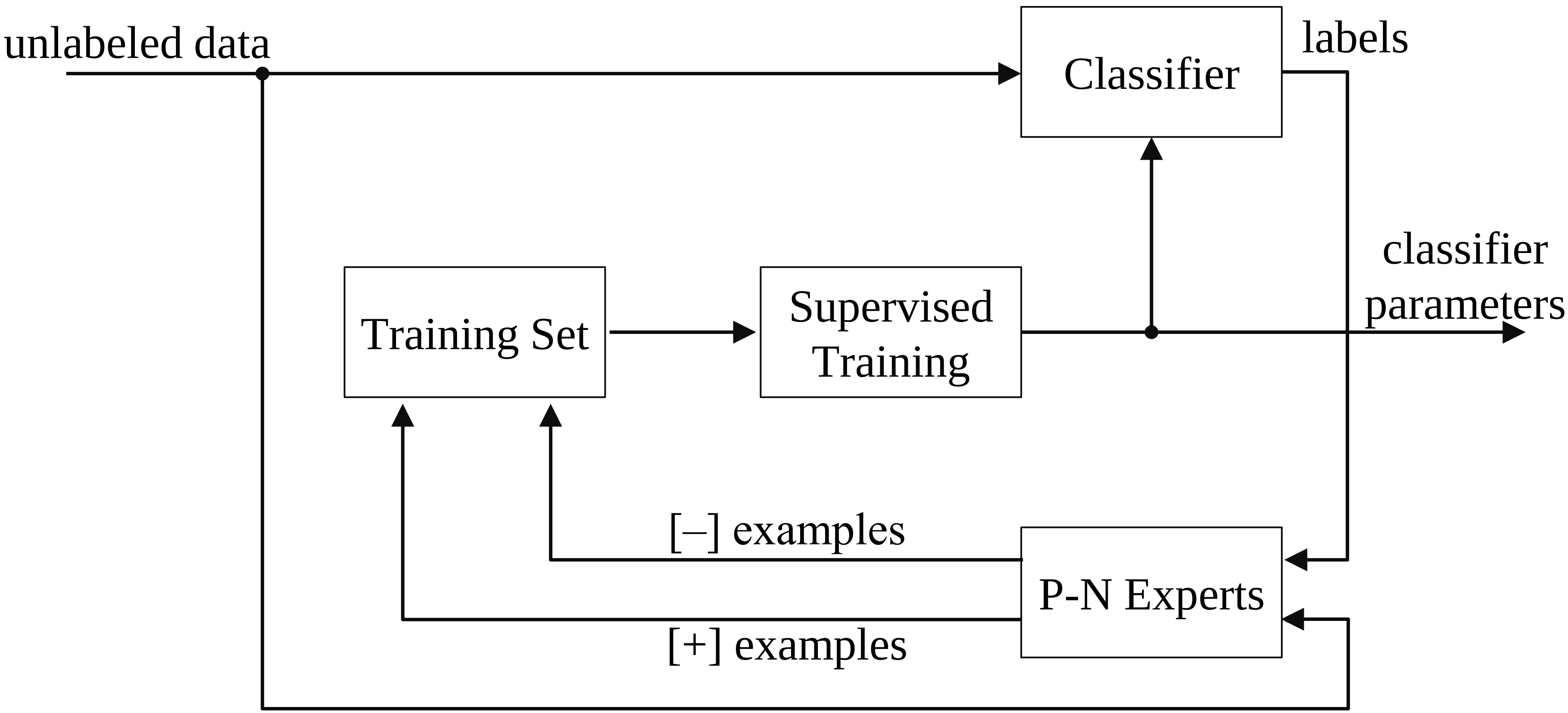


PN – LEARNING: INITIAL LEARNING



TLD FRAMEWORK LEARNING

PN – LEARNING



TLD FRAMEWORK LEARNING

PN – LEARNING